Origen de Java

Características de Java

Las características principales que nos ofrece Java respecto a cualquier otro lenguaje de programación, son:

- ✓ ***Simple***
- ✓ ***Orientado a objetos***
- ✓ ***Permite definición de clases***

¿Qué es una clase?

"Es una abstracción de la realidad, que define una generalidad o colección de elementos con características y comportamientos similares"

Clase Automóvil{// conjunto

Atributos
Peso:
NúmeroRuedas:
Color:
NúmeroPatente:

Métodos:
Arrancar()
Frenar()
Girar()
}

¿Qué es un objeto?
"Un objeto es una instancia o caso particular de una clase".

¿Cómo se crean los objetos?

A través de una operación llamada INSTANCIACIÓN

Instanciación de objetos

Automóvil miFiat, miFord; → Casos particulares de la clase Automóvil

Arquitectura neutral

Java proporciona, pues:

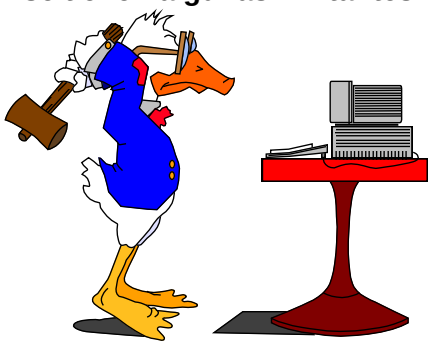
Comprobación de punteros
Comprobación de límites de arrays
Excepciones
Verificación de byte-codes

Seguro

Portable

Interpretado

Todo esto suena muy bonito pero tambien se tienen algunas limitantes:



⇒ La velocidad.
⇒ Los programas hechos en Java no tienden a ser muy rápidos, supuestamente se está trabajando en mejorar esto.Como los programas de Java son interpretados nunca alcanzan la velocidad de un verdadero ejecutable.
⇒ Java es un lenguaje de programación. Esta es otra gran limitante, por más que digan que es orientado a objetos y que es muy fácil de aprender sigue siendo un lenguaje y por lo tanto aprenderlo no es cosa fácil. Especialmente para los no programadores.
⇒ Java es nuevo. En pocas palabras todavía no se conocen bien todas sus capacidades.

Comentarios

En Java hay tres tipos de comentarios:

```
// comentarios para una sola línea

/* comentarios de una o
   más líneas
*/

/** comentario de documentación, de una o más líneas
*/
```

Identificadores

Palabras clave

Las siguientes son las palabras clave que están definidas en Java y que no se pueden utilizar como indentificadores:

abstract	continue	for	new	switch
boolean	default	goto	null	synchronized
break	do	if	package	this
byte	double	implements	private	threadsafe
byvalue	else	import	protected	throw
case	extends	instanceof	public	transient
catch	false	int	return	true
char	final	interface	short	try
class	finally	long	static	void
const	float	native	super	while

Palabras Reservadas

Además, el lenguaje se reserva unas cuantas palabras más, pero que hasta ahora no tienen un cometido específico. Son:

cast	future	generic	inner
operator	outer	rest	var

Operadores

Los operadores de Java son muy parecidos en estilo y funcionamiento a los de C. En la siguiente tabla aparecen los operadores que se utilizan en Java, por orden de precedencia:

. [] ()
++ --
! ~ instanceof
* / %
+ -
<< >> >>>
< > <= >= == !=
& ^ |
&& ||
? :
= op= (*= /= %= += -= etc.) ,

Tipos de dados

Enteros:

byte	8 bits	complemento a dos
short	16 bits	complemento a dos
int	32 bits	complemento a dos
long	64 bits	complemento a dos

Reales en coma flotante:

float	32 bits	IEEE 754
double	64 bits	IEEE 754

Booleanos:

true
false

Caracteres:

Por ejemplo: a \t \u???? [????] es un número unicode

Cadenas:

Por ejemplo: "Esto es una cadena literal"

Arrays

Se pueden declarar en Java arrays de cualquier tipo:

```
char s[];
int iArray[];
```

Incluso se pueden construir arrays de arrays:

```
int tabla[][] = new int[4][5];
```

Los límites de los arrays se comprueban en tiempo de ejecución para evitar desbordamientos y la corrupción de memoria.

En Java un array es realmente un objeto, porque tiene redefinido el operador []. Tiene una función miembro: length. Se puede utilizar este método para conocer la longitud de cualquier array.

```
int a[][] = new int[10][3];
a.length;        /* 10 */
a[0].length;     /*  3 */
```

Para crear un array en Java hay dos métodos básicos. Crear un array vacío:

```
int lista[] = new int[50];
```

o se puede crear ya el array con sus valores iniciales:

Cadenas

```
String nombres[] = {
        "Juan","Pepe","Pedro","Maria"
        };
```

Esto que es equivalente a:

```
    String nombres[];
    nombres = new String[4];
    nombres[0] = new String( "Juan" );
    nombres[1] = new String( "Pepe" );
    nombres[2] = new String( "Pedro" );
    nombres[3] = new String( "Maria" );
```

No se pueden crear arrays estáticos en tiempo de compilación:

```
    int lista[50];  // generará un error en tiempo de compilación
```

Tampoco se puede rellenar un array sin declarar el tamaño con el operador new:

```
    int lista[];
    for( int i=0; i < 9; i++ )
        lista[i] = i;
```

Es decir, todos los arrays en Java son estáticos. Para convertir un array en el equivalente a un array dinámico en C/C++, se usa la clase vector, que permite operaciones de inserción, borrado, etc. en el array.

Control de Flujo
Muchas de las sentencias de control del flujo del programa se han tomado del C:

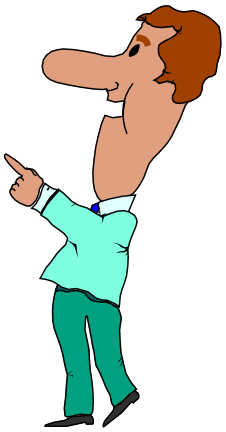
Sentencias de Salto

if/else

```
if( Boolean ) {
   sentencias;
   }
else {
   sentencias;
}
```

switch

```
switch( expr1 ) {
   case expr2:
      sentencias;
      break;
   case expr3:
      sentencias;
      break;
   default:
      sentencias;
      break;
   }
```



Sentencias de Bucle
Bucles for

```
for( expr1 inicio; expr2 test; expr3 incremento ) {
   sentencias;
   }
```

El siguiente trocito de código Java que dibuja varias líneas en pantalla alternando sus colores entre rojo, azul y verde. Este fragmento sería parte de una función Java (método):

```
int contador;
for( contador=1; contador <= 12; contador++ ) {
   switch( contador % 3 ) {
      case 0:
         setColor( Color.red );
         break;
      case 1:
         setColor( Color.blue );
         break;
      case 2:
         setColor( Color.green );
         break;
```

```
        }
    g.drawLine( 10,contador*10,80,contador*10 );
    }
```

También se soporta el operador coma (,) en los bucles for

```
for( a=0,b=0; a < 7; a++,b+=2 )
```

Bucles while

```
while( Boolean ) {
    sentencias;
    }
```

Bucles do/while

```
do {
    sentencias;
    }while( Boolean );
```